THE SERIES 1 INTERFACE ADD-IN FOR TIMEX VERSION OF THE WARAJEVO 2.51 EMULATOR

Note: For those of you that have read this before and don't want to repeat old information (some of it is new), go to the end for a summary of changes in this revision of the Series 1 Interface code and for additional information about the new OS64 version.

This package contains a driver called TSMDR.COM created by Zeljko Juric (with a source file TSMDR.ASM) and two DCK expansion files named TSSER1.DCK and OS64SER1.DCK. These files allow the emulation of the Series 1 Interface while using the Timex mode of the Warajevo 2.51 emulator. If you did not know, the Series 1 Interface is a modification of the normal ZX Interface 1 adapted for use with the Timex Sinclair 2068, which normally does not support this device. Warajevo 2.51 allows the full emulation of the Timex 2068, but unfortunately without the Series 1 Interface. This add-in is dedicated to correct this drawback.

The Timex part of the Warajevo 2.51 emulator does not normally allow the emulation of the microdrive ports because the real Timex has no such device. Fortunately, the Warajevo emulator has the ability to add unimplemented I/O devices using external drivers. If you want details, read more about this in chapter 5 of the Warajevo manual. Source code (fully commented) for TSMDR.COM is supplied in this package for users that want to learn more about making drivers for unsupported I/O devices.

The two DCK files provided in this package contain the modified Timex, OS64 and Series One Interface ROM code necessary to access the emulated microdrives and perform other Interface One commands. To access the emulated microdrives, you must first load the TSMDR.MDR file before starting the Timex part of the emulator. Read the Warajevo manual for more details about selecting and starting the Timex part of the emulation. The simplest way to start the Timex emulator and have microdrive emulation is to type the following command from the command prompt:

```
TS2068 /E<PATH>TSSER1   (with <PATH> equal to DRIVE:\DIRECTORY\)

         or

TS2068 /EOS64SER1       (assuming the DCK file is in the same directory)
```

You can specify the DCK files even if TSMDR.COM is not loaded. However, the emulated microdrives will be reported as not present if you try to access one of the them. BE SURE TO LOAD TSMDR.COM FIRST!

The Timex and Series 1 ROM, packed in the file TSSER1.DCK, have been modified to make the Series 1 code work in the DOCK bank. Since OS64 is normally located in the DOCK bank, the OS64 compatible Series 1 ROM has been modified to work in the HOME bank. The file OS64SER1.DCK contains the necessary ROMS and changes to make the Series 1 ROM work with OS64. One of the big differences between these Series 1 Interface emulations and that of the Spectrum ZX Interface emulation is the location of the Series 1 code and how it is paged into memory. They both get paged into memory by error detection. However, the Spectrum emulation version will get paged into memory by emulated hardware and from a completely independent memory space. The Timex and OS64 versions depend on modified code in the main ROM's and Series 1 ROM's to bank switch the Series 1 ROM's into the processor's address space. This bank switching is completely automatic, and in normal use it is invisible to the user. The DOCK bank should always be enabled, but not necessarily paged into memory, whenever a BASIC program is running or is in the EDIT process. The modified ROM's require the DOCK bank to be enabled during every error report.

There are some other differences between the ZX Spectrum and Timex 2068/OS64 Series 1 systems besides the way the Series 1 ROM is paged into memory. The Spectrum ZX Interface 1 code will add 58 extended variables just below the start of BASIC. This causes the start of BASIC to move up to accommodate these variables. Contrasting with this, this Timex and OS64 emulation will not cause the start of BASIC to move up for the extended variables. All of the extended variables, except one, will be created in the second chunk of the DOCK bank where they will not interfere with BASIC at all. One extended variable, FLAGS3, is located with the other normal Timex variables. There's more about FLAGS3 in the next paragraph. Using the extended variables in the DOCK bank is new with this version of the DCK file. In the previous version, these extended variables were located between the Timex variables and the start of the second display file area. They are now located in the DOCK bank to make more space available for your programs and provide a more secure location for these variables. The Series 1 code does, however, move BASIC up temporarily when accessing the microdrives. A 595 byte buffer is created below BASIC, but, is reclaimed immediately after most microdrive operations.

There is one other item of note for users of the ZX Spectrum. The FLAGS3 variable is not located with the other extended variables. Instead, FLAGS3 is located at address 23732, the same as the low byte for the Physical Ramtop variable. FLAGS3 is available for access only during the time the Series 1 ROM is paged in. At any other time, the value will be maintained at 255. The reason is simple but complicated to explain. Simply put, there weren't enough spare variable locations within the IY registers' relative jump range. Instead address 23732 will do double duty for both the Physical Ramtop and FLAGS3 variables. There shouldn't be any problem unless you get an error report of "Ramtop no good". Even so, it should rarely occur. If it does, execute the command again. If that also fails, try doing the following:

```
POKE 23732,255: POKE 23733,255
```

Try your command one more time. If this error report does happen, the Series 1 ROM did not page out of memory in the normal manner...

The microdrive files for these emulations are the same type as and are compatible with Gerton Lunter's MDR files. The only stipulation is that the

MDR files for use in this emulation need to be named DRIVE1.MDR, DRIVE2.MDR, and so on up to DRIVE8.MDR (DRIVE1.MDR for microdrive 1, etc.). The user can not change the names of microdrive files because they are fixed. The reason for this is that the TSMDR.COM driver must use as little memory as possible. Using fixed names for the microdrive files helps conserve the available memory. If you only want to use microdrive 1, there is no need for all 8 MDR files to exist. The files DRIVE2.MDR through DRIVE8.MDR need not be present.

One of the advantages of having the Series 1 ROM code operate in one of the native banks of the Timex is that we can now LOAD/SAVE programs to/from either the HOME bank or DOCK bank. It is even possible to have a mix of HOME and DOCK bank chunks to SAVE from and LOAD to. If you want to SAVE memory from mixed chunk specifications, you must be careful when loading a previously saved file because the chunk specification will not be saved with the file. You are on your own when loading a file back to its original location. For example, it is possible to save a DOCK bank binary file, 100 bytes long at address 53744, and later, mistakenly load it back to address 53744 in the HOME bank. The Series 1 code will not load the file back to the same memory chunk specification unless you first write the correct value to port 244. To learn what the memory chunk specification is before a save is performed, enter PRINT IN 244. This will return the value that was sent out port 244. Record this value for future reference. When you load the file again, use the value in the command, OUT 244,x, with x equal to the value that was read in from port 244 at the time of the file save. If you plan to use only HOME bank memory, you don't need to be concerned with this information. However, this is a good way to save AROS type programs and load them quickly.

The following math is an example of how to page in the fifth, sixth, and seventh chunks in the DOCK or XROM banks depending on the value written to port 255. DOCK bank memory is enabled when the value written to port 255 is less than 128. XROM bank memory is enabled if the value written to port 255 is greater than 127. For the purpose of determining the paged in chunks, the chunks are numbered zero through seven.

```
OUT 244,(2^4+2^5+2^6)  or  OUT 244,112
```

This emulation cannot save, nor load programs directly from or to the XROM bank. If you have a need for this, use the HOME bank as a temporary buffer and then transfer the memory to the XROM bank by machine code. These comments refer to binary type files only.

You should not page in chunks 0, 1, 2, or 3 of the DOCK or XROM banks when using BASIC commands of the Timex emulator. If using the OS64 program, this list should be chunks 2, 3 and 7 of the DOCK bank and 0, 1, 2, 3, and 7 of the XROM bank. If you do, your program will likely crash. The BASIC interpreter is in the first 16k of the HOME bank (the first 16k of the DOCK bank with OS64). The video memory and the variables are located in the third chunk of the HOME bank and the stack and ram resident code are located in the forth chunk of the HOME bank (in OS64, the stack and ram resident code are located in the eighth chunk or chunk 7). The only safe way to page in these chunks is with machine code, and they must be paged out of memory before returning control to BASIC.

Timex created the ram resident code as a way to communicate with the two other built-in banks of memory and future banks of 64k memory that were supposed to be attached to the computer's expansion bank system. To my knowledge, this expansion bank system was never available.

The TSSER1.DCK file is configured to have ROM type memory in the first 16K of the HOME bank and RAM type memory in the remaining 48K HOME bank address space (as in a real Timex). The DOCK bank memory in this DCK file is configured for 8K of ROM in the first chunk of memory and RAM type memory in the remaining 56K of address space.

The OS64SER1.DCK ROM/RAM space is configured differently than TSSER1.DCK. The entire HOME bank is configured for RAM type memory. The DOCK bank memory is configured for 32K of ROM in the first through the forth chunks and RAM type memory in the remaining 32K of address space. The XROM bank is configured for ROM type memory in the first, third and forth chunks of memory and RAM type memory in the remaining 40k of address space.

If you desire a different ROM/RAM configuration, please read chapter 7.9.4. of the manual that was supplied with the Warajevo 2.51 emulator. It will give you information on the BANK and READ/WRITE header specifications in a DCK files. If you want the Timex or Spectrum ROM space to be configured as RAM, please remember that both the Timex and Spectrum ROM's have a bug that will cause parts of the ROM's to be over-written in certain EDIT operations. To prevent this from happening with the Timex ROM, change the byte at address 2151 decimal from 24 to 23 decimal. The same change should be made in the Spectrum ROM code at address 3372 decimal. It is recommended that the OS64 DOCK bank chunks should be configured only as ROM. If set for RAM type memory, OS64 may not initialize automatically as a LROS program. This happens when the LROS five byte configuration information, beginning at address zero, is changed to zeros or other values. This same problem can occur in the real Timex system as well. The only fix is to rewrite the correct LROS configuration values to the first five bytes just before executing a NEW command.

The file TSSER1.DCK is a combination of three DCK files combined together to form one file. The first file is the Timex HOME bank ROM with a 9 byte header at the beginning (to describe memory configuration). The second file is the Series 1 ROM with a similar 9 byte header at the beginning. The third file is new for this Timex and OS64 emulation. The code contained in this file will be located in the second DOCK bank chunk and will be enabled every time the Series 1 code is paged into memory. This second DOCK bank chunk is configured for RAM type memory because the Series 1 extended variables are located there. More about this later. The other routines located there give the Timex and OS64 extra commands that were provided in the Spectrum Shadow ROM Disassembly book by Gianluca Carri. These commands and their syntax will be listed at the end of this text. To learn how to create your own DCK files, refer to chapter 3.5.2

SUBMENU 'CONVERT' in the manual. The operation of OS64SER1.DCK is too complicated to explain here. However, it is a combination of three nine byte headers, one for each bank of memory, the OS64 ROM, the Timex 2068 ROM, the XROM, the Series 1 ROM and the extended commands ROM.

Most European users of the ZX Spectrum are familiar with the microdrive syntax. However, very few of the USA users know this syntax. For those that do not know the syntax, refer to the following list of commands (not fully complete, but useful for most users):

TIMEX SERIES 1 SYNTAX	COMMENTS
FORMAT "m";n;"name"	This must be done to create cartridge.
CAT n	n = microdrive number, 1 through 8.
LOAD *"m";n;"file"	"m" = microdrive channel.
LOAD *"m";n;"file"CODE	Load a binary file.
LOAD *"m";n;"file"CODE x	Load a binary file starting at address x.
LOAD *"m";n;"file"SCREEN$	Load SCREEN file.
SAVE *"m";n;"file"LINE x	Auto starting program at line x.
SAVE *"m";n;"file"CODE x,y	Save y bytes at address x.
SAVE *"m";n;"file"SCREEN$	Save SCREEN file, primary display only.
SAVE *"m";n;"file"CODE 24576,6912	Save second display file, if enabled.
MOVE "m";n;"file" TO #x	Move data file to stream x.
MOVE "m";n;"file" TO "m";n;"file2"	Copy data file to file2.
ERASE "m";n;"filename"	Obvious, but don't add CODE for bin files!
VERIFY *"m";n;"filename"	Verify RAM contents match mircodrive's.
MERGE *"m";n;"filename"	Works on HOME bank memory only!!!
OPEN #n;"m";x;"file"	Open a data file for reading or writing.
CAT #n;x	Catalog of Microdrive x sent to stream n.
CLS #	Restore the normal BORDER, PAPER, and INK.
CLEAR #	Close all streams and reclaim memory.
CLOSE #n	Close stream n; reclaim memory used by it.
CLEAR #n	For Timex only. Does the same as CLOSE #n.
INPUT #n;...	Read string in from stream n.
INKEY$ #n	Read character in from stream n.
PRINT #n;...	Output print sequence (...) to stream n.

NOTE: PRINT #n is a normal Timex command which is the same as LPRINT when used as in PRINT #3. Even so, if we tried to OPEN 3#;"t" in the previous Timex emulation, data would be sent to a non-supported serial port. Now, in these new Timex/OS64 versions, data will instead be sent to an emulated AERCO printer interface.

Other Series 1 commands exist, but they are not functional in this emulation due to non-supported I/O ports, like:

FORMAT "n";x	Set the network station number to x.
FORMAT "t";x or FORMAT "b";x	Set the serial port baud rate.

LOAD *″t″ or LOAD *″b″	Commands for the Series 1 serial port.
SAVE *″t″ or SAVE *″b″	Commands for the Series 1 serial port.
LOAD *″n″;n or LOAD *″n″;nCODE	Commands for the Series 1 network.
SAVE *″n″;n or SAVE *″n″;nCODE x,y	Commands for the Series 1 network.
OPEN #n;″x″	Open binary ″b″, text ″t″, or net ″n″ strm.

NOTE: The separator character in, OPEN #n;″x″, must be a ″;″ in Timex mode even though a ″,″ is allowed in the Spectrum ZX Interface 1. The comma passes the Timex syntax check and does not page in the Series 1 ROM. If used, all you will get is an ″Invalid I/O device″ error report.

SUMMARY OF CHANGES TO TIMEX SERIES 1 DCK AND NEW OS64 SERIES 1 FEATURES

1. 57 of the 58 extended variables for the Series 1 interface have been moved to the DOCK bank starting at address 2002H. (Timex and OS64)

2. The second DOCK bank chunk now has extra command routines that are enabled with OPEN #3;″f″. The extra commands include all of the BASIC commands that Gianluc Carri listed in his book, Spectrum Shadow ROM Disassembly. (Second DOCK bank chunk in Timex; second HOME bank chunk in OS64)

   a. * CAT n   A full more complete CATalog listing.

   b. POKE *X,Y   Allows a simple double poke of a 16 bit number to two consecutive addresses.

   c. POKE X,″string″   POKE string to memory starting at address X.

   d. *Ln   (mod of book ver)   Memory dump starting at decimal address n. The OS64 version will display 16 addresses and their ASCII characters. It can be changed to 14 address values or 8 address address values by: LPRINT ?14 or LPRINT ?8. Change it back to 16 by: LPRINT ?16. Addresses from 0 through 16383 will always display addesses for the Series 1 ROM, and the address space for the new commands and the extended variables.

   e. LPRINT ?8 (mod of book ver)   See 2d.

   f. LPRINT ?14 ″ ″ ″ ″   See 2d.

   g. LPRINT ?16 ″ ″ ″ ″   See 2d.

   h. LPRINT ?1   ″ ″ ″ ″   Opens stream #3 to use AERCO printer driver located in second DOCK chunk. This is not the same driver code as in Carri’s book.

i. LPRINT ?n ″ ″ ″ ″ Closes stream #3 and restores original stream data. Use any value for n other than 1, 8, 14, and 16.

j. LPRINT !n ″ ″ ″ ″ Sets maximum printer line width. Relocated from address 5CB1H to 2001H. Address 2000H is new location for the printer char. pos.

k. *En Modified EDIT function. Immediately moves line n to the edit line.

l. BEEP *a,b,c,d An improved sound command. Try entering the following as a test: BEEP *100,255,10,1

m. READ #S,N Provides ability to do pseudo-random file handling. When READ #S,N is executed, the 'read' microdrive channel attached to the stream 'S' is used, and record 'n' of that file is loaded into the channel buffer. Read the book by Gianluc Carri for a more detailed explanation.

n. RESTORE #S Provides apparent ability to write to a READ type file and add additional data to it. Read the book by Gianluc Carri for a more detailed explanation.

3. Channels B and T have been modified to LPRINT and LLIST to an emulated AERCO printer driver. To enable the driver do: OPEN #3;″t″. Currently there seems to be no advantage in sending the data directly to channel B even though the channel B code has been modified for the AERCO interface printer port. (Timex and OS64)

4. The program, Keyword version 6, by Jack Dohany, is located in the second chunk of the DOCK bank (Timex version only). Keyword allows the keywords to be entered one character at time in upper or lower case. It is not run from the DOCK bank but will be relocated to the an area between the end of the Timex variables and the start of the second display file. When enabled with OPEN 3;″f″, the start of BASIC permanently moves up by 11 bytes to make room for this extended channel and another 11 bytes for channel T. If channel T is not wanted, do: CLOSE #3 or CLEAR #3. The normal Keyword program does not need 11 bytes for the channel, but the Series 1 code would not work correctly without exactly 11 bytes created for this channel. Once created, it cannot be be closed without doing a NEW command. It should also be the first channel created. Only four characters are required to be entered in order for the keyword to be recognized. Only two keywords, OPEN # and CLOSE #, are entered differently than the other keywords. For these two keywords, enter only OPEN or CLOSE and without the ″#″ sign.

5. OS64 is located in the first two DOCK bank chunks. The Series 1 code is located in the first HOME bank chunk. The extra command routines are located in the second HOME bank chunk along with the Keyword version 6 code and an AERCO printer driver. Other than this, the information in #4 above applies to OS64 as well.

6. The OS64 flash routine has been modified to give an inverse video cursor instead of the true video cursor. The error position cursor in the edit line will also be inverse video in this version.

7. The OS64 default printer driver, in this version, is now the AERCO driver. In other versions of the OS64 LROS code, you might have to poke address 65523 to value 167 to enable the AERCO driver. This version does not need the poke change to enable the AERCO driver. The Series 1 ROM version of the AERCO printer driver can be enabled by doing OPEN #3;"t". The second chunk AERCO driver can be enabled by OPEN #3;"f". The printer line width in the default AERCO driver is changed by poking address 65525 decimal with the desired line length. The printer line width for channel T and F are set with LPRINT !n. Another way is to use machine code to access address 2001H in the DOCK bank. The printer line width and printer character position were changed from 5CB1H and 5CB0H because the OS64 LROS program used these same spare variable locations.

8. The user can load and save machine code directly in the DOCK bank address space from 2000H through 3FFFH (8192 through 16383 decimal). This address space is always enabled during any Series 1 ROM access. There is some code already there so it is best that you restrict your code to certain areas. Currently, there is free space available for the user's own code from 2100H through 2FFFH and 3D00H through 3FFFH. If you have knowledge in accessing the extra banks of Timex memory and know how to add additional commands to work with the Series 1 ROM code, these areas are good locations to put your own code. These areas are less likely to interfere with other programs in the BASIC programming area and obviously will not take up any space in the BASIC programming area.

NOTE: a. The COPY command in OS64 is designed for use with EPSON type printer control codes!

b. The program Super HotZ is not compatible with the Keyword program. Both programs use some of the same address locations between the Timex variables and the start of the second display file location.

c. The program Super HotZ is not compatible with this version of the OS64 emulation. Super HotZ uses the Timex HOME bank ROM routines while it is in operation. The Timex HOME bank ROM does not exist in this emulation except for a brief time after a NEW command. After which, it is immediately overwritten with the Series 1 ROM code.